REMARKS

Attorney for Applicant has carefully reviewed the outstanding Office Action on the above-identified application. Applicant has amended the application as set forth herein, and submits that the application, as amended, is in condition for allowance.

Applicant has amended claim 1 to better define Applicant's claimed invention. Specifically, claim 1 was amended to recite that the mobile agents of the present invention are executed using execution code **provided to the mobile agents from a central server,** and to include the step of **maintaining stack trace and state information about each of the mobile agents to allow one or more of the mobile agents to be reconstructed at an alternate computing host using the stack trace and state information.** Additionally, Applicant has amended claim 3 to provide antecedent basis in view of Applicant's amendments to claim 1. Further, Applicant has amended claim 9 to correct a minor typographical error.

Applicant has amended claim 11 to better define Applicant's claimed invention. Specifically, claim 11 was amended to recite the steps of assigning **a computing task to one or more mobile agents**; transferring the **one or more** mobile agents to **one or more available** computing hosts; transferring execution code from a central server to the **one or more available** computing hosts; executing the **one or more** mobile agents at the **one or more available** computing hosts using the execution code; and storing **stack trace** and real-time state information about the **one or more** mobile agents **at a first alternate computing host on which none of the one or more mobile agents are executing to allow the one or more mobile agents to be reconstructed at a second alternate computing host.**

Applicant has amended claim 14 to better define Applicant's claimed invention. Specifically, the preamble of claim 14 was amended to recite a method for migrating a **software application running in a virtual machine**, and the body of the claim was amended to recite the steps of constructing an application using a **plurality of mobile agents; transferring the plurality of mobile agents to a first computing host; executing the mobile agents at the first computing host; maintaining stack trace and state information about each of the plurality of mobile agents at a second computing host on which none of the plurality of mobile agents are executing;** detecting an indication to migrate the application; and in response to the indication, migrating the application **in its entirety** from the first computing host to a **third computing host** without modifying a virtual machine **at the third computing host by reconstructing each of the plurality of mobile agents at the third computing host using the stack trace and state information.** Additionally, in view of Applicant's amendments to claim 11, Applicant has amended claims 20 and 21 to provide antecedent basis and has cancelled claims 22-24.

Applicant has amended claim 26 to better define Applicant's claimed invention. Specifically, the preamble of the claim was amended to recite a heterogeneous **computing environment**, and the body of the claim was amended to recite computing resources on a network including **virtual machines for executing mobile agent software code**; and means for transferring execution code from a central server to the computing resources, the computing resources executing one of the small tasks assigned to a mobile agent in the virtual machines using the execution code **and the means for transferring execution code maintaining stack trace and state information about each of the mobile agents at a first alternate computing**

host where none of the mobile agents are executing to allow each of the mobile agents to be reconstructed at a second alternate computing host. In view of Applicant's amendments to claim 26, Applicant has cancelled claim 27.

Applicant has amended claim 34 to better define Applicant's claimed invention. Claim 34 was amended to recite the steps of storing **stack trace and state** information about the mobile agent thread **at a second computing host at which the mobile agent thread is not executing** as the mobile agent thread executes **at the first computing host; transferring the execution code for the mobile agent thread from a central server to a third computing host**; and transferring **stack trace and state information** about the mobile agent thread to the **third computing host.** In view of the amendments to claim 34, Applicant has cancelled claims 35-37, and has amended claims 38 and 39 to provide antecedent basis with amended claim 34.

Applicant has amended claim 40 to better define Applicant's claimed invention. Specifically, claim 40 was amended to recite **an agent debugger for storing stack trace and state information about each of the plurality of mobile agents at a computing host where none of the plurality of mobile agents are executing.**

Applicant has added new claims 42-45 to further define Applicant's claimed invention. Claims 42-45 each recite a method for dynamically constructing and executing a computer application in a heterogeneous computing environment using mobile agents comprising the steps of **constructing a computer application using mobile agents; storing the mobile agents at a central server; dispatching the mobile agents from the central server to an available**

computing host on the heterogeneous computing environment using a dispatching component; allowing the mobile agents to execute at the available computing host; and monitoring execution of each of the plurality of mobile agents and storing stack trace and state information about execution of the mobile agents at a computing host where none of the mobile agents are executing. Further, Applicant has added dependent claims 46-51 to further define Applicant's claimed invention. No new matter is believed to have been introduced by any of the aforementioned amendments.

Applicant's claimed invention provides a method and apparatus for providing parallel execution of computing tasks in heterogeneous computing environments using mobile agents. The method includes the steps of partitioning a computing task into small tasks; assigning the small tasks to mobile agents; determining available computing hosts in the heterogeneous computing environment; transferring the mobile agents to the available computing hosts; executing the mobile agents at the available computing hosts using execution code provided to the mobile agents from a central server; and maintaining stack trace and state information about each of the mobile agents to allow one or more of the mobile agents to be reconstructed at an alternate computing host using the stack trace and state information. (Claim 1)

Applicant's claimed invention further provides a method for providing parallel computing using mobile agents, comprising the steps of: assigning a computing task to one or more mobile agents; transferring the one or more mobile agents to one or more available computing hosts; transferring execution code from a central server to the one or more available computing hosts; executing the one or more mobile agents at the one or more available computing hosts using the execution code; and storing stack trace and real-time state information about the one or more mobile agents at a first alternate computing host on which none of the one or more mobile agents are executing to allow the one or more mobile agents to be reconstructed at a second alternate computing host. (Claim 11)

Applicant's claimed invention further provides a method for migrating a software application in a virtual machine from a primary host to a secondary host. The method comprises the steps of: constructing an application using a plurality of mobile agents; transferring the plurality of mobile agents to a first computing host; executing the plurality of mobile agents at the first computing host; maintaining stack trace and state information about each of the plurality of mobile agents at a second computing host on which none of the plurality of mobile agents are executing; detecting an indication to migrate the application; and in response to the indication, migrating the application in its entirety from the primary host to a third computing host without modifying a virtual machine at the third computing host by reconstructing each of the plurality of mobile agents at the third computing host using the stack trace and state information. (Claim 14)

Applicant's claimed invention also provides a computer system apparatus for providing parallel execution of computing tasks in a heterogeneous computing environment. The system includes: a dispatcher for partitioning the computing task into a plurality of small tasks and dispatching the small tasks; mobile agents for receiving small tasks from the dispatcher; computing resources on a network including virtual machines for executing mobile agent software code; means for transferring the mobile agents to the computing resources; and means for transferring execution code from a central server to the computing resources, the computing resources receiving and executing one of the small tasks assigned to a mobile agent in the virtual machines using the execution code and the means for transferring execution code maintaining stack trace and state information about each of the mobile agents at a first alternate computing host where none of the mobile agents are executing to allow each of the mobile agents to be reconstructed at a second alternate computing host. (Claim 26)

Applicant's claimed invention further provides a method for providing realistic thread migration. The method comprises the steps of: instantiating a mobile agent thread at a first computing host; processing the mobile agent thread at the first computing host; storing stack trace and state information about the mobile agent thread at a second computing host at which the mobile agent thread is not executing as the mobile agent thread executes at the first computing host; detecting an indication to migrate the mobile agent thread; and in response to the indication, stopping execution of the mobile agent thread; transferring the execution code for the mobile agent thread from a central server to a third computing host; and transferring the stack trace and state information about the mobile agent thread to the third computing host. (Claim 34)

Applicant's claimed invention provides an agent collaboration environment. The environment comprises: a plurality of mobile agents; an agent debugger for storing stack trace and state information about each of the plurality of mobile agents at a computing host where none of the mobile agents are executing; a conference room for providing a virtual workspace for the mobile agents; and a registration subsystem for selectively assigning the plurality of mobile agents to the conference room, wherein each of the plurality of agents can share data, information, and results of computations in the conference room. (Claim 40)

Applicant's claimed invention further provides a method for dynamically constructing and executing a computer application in a heterogeneous computing environment using mobile agents. The method comprises the steps of: constructing a computer application using mobile agents; storing the mobile agents at a central server; dispatching the mobile agents from the central server to an available computing host on the heterogeneous computing environment using a dispatching component; allowing the mobile agents to execute at the available computing host; and monitoring execution of each of the plurality of mobile agents and storing stack trace and state information about execution of the mobile agents at a computing host where none of the mobile agents are executing. (Claim 43)

The feature of the present invention of providing execution code to mobile agents from a central server allows for lightweight agents to be used on the network, such that execution code need not be carried by each agent. Further, by maintaining stack trace and state information at an available host other than the one initially executing the agent, or to which the agent subsequently migrates, each agent can be reconstructed using the stack trace and state information at one or more alternate computing hosts, and execution can be resumed at such hosts precisely where it left off. Moreover, this feature allows each agent to execute on virtual machines at each node without requiring modification of the virtual machine at each node, a significant advantage not found in the prior art. Importantly, the present invention provides immunity to computer or network hardware, power, and communications failures, such that if power or communications is lost at a node on which an agent is executing, or a portion of a computer network is destroyed by an unforeseen catastrophic event (such as by a fire or earthquake), the agent can still be reconstructed at another node using the stack trace and state information that has already been stored. This allows executing threads and applications to be seamlessly migrated from one host to another, regardless of power, hardware, or communications failures.

Applicant submits that the pending claims, as amended herein, are patentable over U.S. Patent No. 6,330,583 to Reiffin and U.S. Patent No. 6,587,432 to Putzolu, et al., taken alone or in combination.

Reiffin discloses a computer network of interactive multitasking computers for parallel processing of network subtasks concurrently with local tasks. The network comprises a plurality of workstations or personal computers (PC's), each of which have pre-emptive multitasking for the interactive execution of local tasks concurrently with remote network subtasks. One of the workstations or PC's includes a network disk drive that constitutes a pool for storing all of the subtask identifiers, preferably in the form of a queue (*see* col. 2, lines 22-25). Each workstation or PC is periodically interrupted (after a timeslice of about 20 milliseconds), and a remote network subtask identifier is fetched from the pool and the corresponding subtask is copied from an originating computer to the workstation or PC and executed thereby (*see* col. 2, lines 34-47). After execution of the remote subtask, control is returned to the workstation or PC for execution of local tasks or additional processing of remote network subtasks (*see* col. 2, lines 48-67).

Putzolu, et al. discloses a method and system for diagnosing network congestion using mobile agents. The invention analyzes traffic on a network, detects congestion, and gathers information about the traffic by launching an agent and having the agent iteratively identify which of the links on the node on which the agent operates accepts a typc or class of traffic. The agent can traverse the identified link to the node across the network, and can repeat the process. A monitoring agent generates the trace agent, dispatches it to a designated node, waits for it to report back, destroys the agent, and then selects a new node to investigate, repeating the process (*see* col. 4, lines 34-38). The trace agents are capable of executing on a node in the network, stopping execution, transporting themselves to other nodes, and resuming execution (*see* col. 4, lines 7-10).

Applicant submits that neither Reiffin nor Putzolu, et al., taken alone or in combination, teach or suggest each element of Applicant's claimed invention, as set forth in amended independent claim 1. Specifically, neither Reiffin nor Putzolu, et al., taken alone or in combination, teach or suggest executing mobile agents at available computing hosts teach or suggest **maintaining a stack trace and state information about each of the mobile agents to allow one or more of the mobile agents to be reconstructed at an alternate computing host using the stack trace and state information**, as set forth in claim 1. Reiffin is wholly unconcerned with using mobile agents to allow for the execution of computing tasks on one or more hosts, much less maintaining stack trace and state information so that the agents can be reconstructed at onc or more alternate hosts. Rather, Reiffin merely provides a network directory wherein one or more network subtasks can be fetched from a pool and executed locally by a workstation or PC having a pre-emptive multitasking operating system.

Putzolu, et al. fails to remedy the deficiencies of Reiffin, and is likewise absent any teaching, suggestion, or motivation to provide execution of computing tasks using mobile agents, wherein stack trace and state information is stored so that the agents can be reconstructed at one or more alternate hosts. As a threshold matter, Putzolu, et al. is unconcerned with utilizing mobile agents to allow computing tasks to be executed on one or more remote hosts. Rather, Putzolu, et al. merely uses agents for tracing network congestion, and not for executing computing tasks on one or more remote hosts. While Putzolu, et al. discloses that each agent can carry state information with it as the agent migrates across the network, Putzolu, et al., is devoid of any teaching, suggestion, or motivation relating to storing stack trace and state information so that the agents can be reconstructed at a second host using the stack trace and state information. Indeed, although the agents of Putzolu, et al. can migrate, because the system of Putzolu, et al. does not maintain stack trace and state information, the agents cannot be reconstructed at other nodes and execution continued at such nodes exactly where such execution left off on the original host.

Unlike the work of either Putzolu, et al. or Reiffen, the present invention can support the virtual migration of a running thread across a heterogeneous computer network. Additionally, because the debugger component of the present invention runs on a host machine other than a host machine on which an agent initially executes or is migrated to, the present invention can support seamless migration even in the event of a computing node, network, or power failure. Accordingly, Applicant submits that amended claim 1 and claims 2-10, which depend from claim 1 and contain all of the limitations thereof, are patentable over Reiffin in view of Putzolu, et al., taken alone or in combination.

Applicant submits that claim 11, as amended herein, is patentable over Reiffin in view of Putzolu, et al. Neither of these references, taken alone or in combination, teach or suggest **storing stack trace and real-time state information about one or more mobile agents at a first alternate computing host on which none of the mobile agents are executing to allow the one or more mobile agents to be reconstructed at a second alternate computing host**, as set forth in amended claim 11. Reiffin, discussed earlier, is absent any teaching, suggestion, or motivation to use mobile agents to allow for the execution of a computing task on one or more remote hosts, much less storing stack trace and state information about each mobile agent at a computing host on which none of the mobile agents are executing. Similarly, while Putzolu, et al. employs mobile agents to detect network congestion, Putzolu, et al. is absent any teaching, suggestion, or motivation to use mobile agents for executing computing tasks on one or more remote hosts, or storing stack trace and state information about each mobile agent at a computing host where none of the mobile agents are executing so that the agents can be reconstructed at alternate hosts.

Additionally, the state information stored Putzolu, et al. is incapable of being utilized to reconstruct a mobile agent at an alternate host. For example, Putzolu, et al. merely stores member data, work objects, worksheets, and an access control list (*see* col. 9, lines 64-67). Further, Putzolu, et al. discloses that the state information may also include data created when an agent is instantiated or after it is created (*see* col 10 lines 1-6). However, Putzolu, et al. fails to teach or suggest maintaining an execution stack trace, which is vital information about a running thread and which is essential to the process of reconstructing the thread at an alternate host. Further, Putzolu, et al. does not disclose using mobile agents to allow for the execution of a software application in a heterogeneous computing environment, but rather, merely uses mobile agents to trace traffic congestion on a network. The present invention, by contrast, allows for the development of complete business software applications that are capable of migrating completely intact across a heterogeneous computer network to alternative hosts while they are executing. As such, Applicant submits that claim 11 and claims 12-13, which depend from claim 11 and contain all of the limitations thereof, are patentable over Reiffin and Putzolu, et al., taken alone or in combination.

For similar reasons, Applicant submits that claim 14 and claims 15-21 and 25 depending therefrom, are patentable over Reiffin in view of Putzolu, et al. Neither of these references, taken alone or in combination, teach or suggest **maintaining stack trace and state information about each of a plurality of mobile agents at a second computing host on which none of the plurality of mobile agents are executing**, nor do these references teach or suggest **migrating an application in its entirety from the primary host to a third computing host without modifying a virtual machine at the third computing host by reconstructing each of the plurality of mobile agents at the third computing host using the stack trace and state information**, as set forth in each of these claims. Accordingly, Applicant submits that claims 14-21 and 25 are patentable over Reiffin and Putzolu, et al., taken alone or in combination.

Applicant submits that neither Reiffin nor Putzolu, et al., taken alone or in combination, teach or suggest each element of Applicant's claimed invention as set forth in amended claim 26. Neither Reiffin nor Putzolu, et al. teach or suggest **transferring execution code from a centralized server to computing resources for use thereat by mobile agents**. To the contrary, Putzolu, et al. teaches that agent bytecodes are transferred directly from one proactive environment to another, using facilities for object serialization that are built into the Java virtual machine itself. Putzolu, et al. discloses that Java serialization requires that compiled class files, which are representative of uninstantiated mobile agents, must first be installed in the "Classpath" on the host to which the agent will be migrated (*see* col. 10, lines 1-25). Reiffin similarly fails to teach or suggest transferring execution code from a central server to computing resources for use by mobile agents. Rather, the system of Reiffin is predicated upon "fetching" (requesting) subtasks from a centralized server, and is unconcerned with dispatching tasks to computing resources for use by mobile agents.

Additionally, neither Reiffin nor Putzolu, et al. teach or suggest a computer system apparatus for providing parallel execution of computing tasks in a heterogeneous computing environment, which includes **a dispatcher for partitioning the computing task into a plurality of small tasks** and means for transferring execution code, the means for transferring execution code **maintaining stack trace and state information about each of the mobile agents at a first alternate computing host where none of the mobile agents are executing to allow each of the mobile agents to be reconstructed at a second alternate computing host**, as set forth in amended claim 26. Claims 28-33, which depend from claim 26 and contain all of the limitations thereof, are likewise patentable over Reiffin in view of Putzolu, et al.

Moreover, Applicant submits that neither Reiffin nor Putzolu, et al., taken alone or in combination, teach or suggest each element of Applicant's claimed invention as set forth in amended claim 34 and claims 38-39 depending therefrom, nor do they teach or suggest each element of Applicant's claimed invention as set forth in amended claim 40 and claim 41 depending therefrom. Claims 34 and 38-39 each recite the steps of **storing stack trace and state information about a mobile agent thread at a second computing host at which the mobile agent thread is not executing, transferring execution code for the mobile agent from a central server to a third computing host,** and **transferring the stack trace and state information about the mobile agent thread to the third computing host**. Both Reiffin and Putzolu, et al. are absent such features. Further, claims 40-41 each recite features not found in Reiffin and Putzolu, et al., namely, **an agent debugger for storing stack trace and state information about each of a plurality of mobile agents at a computing host where none of the plurality of mobile agents are executing.** As such, Applicant submits that claims 34, 38-39, and 40-41 are patentable over Reiffin and Putzolu, et al., taken alone or in combination.

Finally, Applicant submits that Reiffin and Putzolu, et al., taken alone or in combination, fail to each or suggest each element of new claim 43 and claims 44-46 depending therefrom. Neither Reiffin nor Putzolu, et al. teach or suggest **monitoring execution of each of the plurality of mobile agents and storing stack trace and state information about execution of the mobile agents at a computing host where none of the mobile agents are executing**, as set forth in these claims.

All issues raised in the Office Action are believed to have been addressed. Claims 1, 3, 9, 11, 14, 20, 21, 26, 34, and 38-40 were amended. Claims 22-24, 27, and 35-37 were cancelled, and claims 42-51 was added. Claims 1-21, 25-26, 28-34, and 38-51 are pending in this application. Re-examination is requested and favorable action solicited.

Respectfully submitted,

[illegible]

Dated: 12/8/04

Michael R. Friscia
Reg. No. 33,884
Attorney for Applicant
Wolff & Samson PC
One Boland Drive
West Orange, NJ 07052
Tel.: (973) 530-2024
Fax.: (973) 530-2224